UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/715,772	11/17/2000	Jack B. Dennis	84781-US2	7033

26111 7590 07/31/2006

STERNE, KESSLER, GOLDSTEIN & FOX PLLC
1100 NEW YORK AVENUE, N.W.
WASHINGTON, DC 20005

EXAMINER	
KING, JUSTIN	
ART UNIT	**PAPER NUMBER**
2111	

DATE MAILED: 07/31/2006

Please find below and/or attached an Office communication concerning this application or proceeding.

UNITED STATES PATENT AND TRADEMARK OFFICE

Commissioner for Patents
United States Patent and Trademark Office
P.O. Box 1450
Alexandria, VA 22313-1450
www.uspto.gov

MAILED

JUL 31 2006

Technology Center 2100

**BEFORE THE BOARD OF PATENT APPEALS
AND INTERFERENCES**

Application Number: 09/715,772
Filing Date: November 17, 2000
Appellant(s): DENNIS ET AL.

Edward J. Kessler
For Appellant

EXAMINER'S ANSWER

This is in response to the appeal brief filed 5/8/2006 appealing from the Office action mailed 8/19/2005.

(1) Real Party in Interest

A statement identifying by name the real party in interest is contained in the brief.

(2) Related Appeals and Interferences

The examiner is not aware of any related appeals, interferences, or judicial proceedings which will directly affect or be directly affected by or have a bearing on the Board's decision in the pending appeal.

(3) Status of Claims

The statement of the status of claims contained in the brief is correct.

(4) Status of Amendments After Final

The appellant's statement of the status of amendments after final rejection contained in the brief is correct.

(5) Summary of Claimed Subject Matter

The summary of claimed subject matter contained in the brief is correct.

(6) Grounds of Rejection to be Reviewed on Appeal

The appellant's statement of the grounds of rejection to be reviewed on appeal is correct.

(7) Claims Appendix

The copy of the appealed claims contained in the Appendix to the brief is correct.

(8) Evidence Relied Upon

No evidence is relied upon by the examiner in the rejection of the claims under appeal.

(9) Grounds of Rejection

The following ground(s) of rejection are applicable to the appealed claims:

Claim Rejections - 35 USC § 102

1. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

2. Claims 1-12, 14-25, 27-38, and 41 are rejected under 35 U.S.C. 102(b) as being anticipated by Motomura (U.S. Patent No. 5,815,727).

Referring to claim 1: As stated by both the Application Specification and the previous response Paper #13, each processing slice executes several threads concurrently in the same clock cycle by interleaving the instructions' fetch, decode, dispatch, and execution (Paper#13, page 2, lines 6-7, Specification, page 8, lines 21-25). Thus, the processing slice executes several threads concurrently by placing the plurality of threads on an execution pipeline; and by

interleaving the threads, the processing slice alternatively processes the instruction from different threads while placing the earlier thread in a waiting state.

Motomura discloses a processing slice (figure 1, structure 100) to execute a plurality of threads. Motomura's processing slice comprises a functioning unit (figure 1, structure 110) to perform a register operation (column 8, lines 1-51). The requestor/generator for the threaded operations is the peripheral device, and the means to convey the request is the peripheral bus. Motomura discloses interleaving threads during execution of a certain thread (column 8, lines 15-27, forking operation), which is placing a plurality of threads in one execution pipeline by interleaving. Thus, Motomuras' processing slice executes a plurality of threads concurrently as disclosed in the Appellant's Specification.

As further stated by both the Application Specification and previous response Paper #13, the claimed limitation "functional unit to perform a register operation specified in the instructions in each of the plurality of threads" directs toward performing an operation specified in the dispatched instruction and performs all operations of the instruction set that manipulate values in the data registers (Paper#13, page 2, lines 8-10, Specification, page 12, lines 5-7). Motomura discloses performing operations dispatched via the bus 111 (column 8, lines 15-23); thus, Motomura's functional unit performs a register operation specified in the instructions in each of the plurality of threads as disclosed in the Appellant's Specification.

Hence, claim is anticipated by Motomura.

Referring to claims 2-3: Any request from the user, such as keyboard inputs or printing, trigs a threaded operation. Such request is an I/O operation, and the device receives the request, such as a keyboard or a mouse, is an I/O device.

Referring to claim 4: Motomura discloses the message content (column 8, lines 36-39).

Referring to claim 5: Once each thread's program counter finishes every associated instruction and is ready for a new thread, the means to ask for new thread's information indicates the completion of the previous thread.

Referring to claim 6: Motomura discloses the processing parameters and the pointer pointing to the parameters (column 8, lines 37-38). The pointer is the data register address.

Referring to claim 7: Motomura's bus is bi-directional (figure 1, structures 111 and 112, column 8, line 16).

Referring to claims 8-9: Motomura discloses executing a different thread while the first thread is in a waiting state (column 8, lines 40-42). Motomura's switching to different thread is the claimed disabling the first thread. Motomura discloses an executing scenario which executing a program sequentially without dividing (column 1, line 61). The disclosed executing a program without dividing is the claimed continuing to execute if in a non-wait instruction.

Referring to claim 10: Motomura discloses resuming the thread previously in a waiting state (column 2, lines 59-60).

Referring to claim 11: Motomura discloses an instruction processing unit (figure 1, structure 150) to process instructions fetched from a program memory (figure 1, structure 140); and a thread control unit (figure 16's thread execution control system) coupled to the instruction processing unit to manage initiating and termination of at least one of the plurality of threads.

Referring to claim 12: Motomura discloses a memory access unit (figure 16, structure 620) coupled to the instruction processing unit to provide access to one of a plurality of data memories (figure 5, structure 541) via a data memory switch (figure 5, structure 542), the

memory access unit having a plurality of data base registers (figure 16, structure 1610), each of the data base registers corresponding to each of the threads; and a functional unit (figure 1, structure 110) coupled to the instruction processing unit to perform an operation specified in one of the instructions; and a register file (figure 16, structure 1420) having a plurality of data registers (figure 16, structure 1410), each of the data registers corresponding to each of the threads.

Referring to claim 14: Motomura discloses a processing slice (figure 1, structure 100) to execute a plurality of threads. Motomura's processing slice comprises functioning unit (figure 1, structure 110) to perform a register operation (column 8, lines 1-51). The requestor/generator for the threaded operations is the peripheral device, and the means to convey the request is the peripheral bus.

As stated by both the Application Specification and the previous response Paper #13, each processing slice executes several instructions concurrently in the same clock cycle by interleaving the instructions' fetch, decode, dispatch, and execution (Paper#13, page 2, lines 6-7, Specification, page 8, lines 21-25). Motomura discloses forking other threads during execution of a certain thread (column 8, lines 15-27), which is the claimed execution, fetch, and decode in the same clock cycle.

As stated by both the Application Specification and previous response Paper #13, the claimed limitation "functional unit to perform a register operation specified in the instructions in each of the plurality of threads" directs toward performing an operation specified in the dispatched instruction and performs all operations of the instruction set that manipulate values in the data registers (Paper#13, page 2, lines 8-10, Specification, page 12, lines 5-7). Motomura

discloses performing operations dispatched via the bus 111 (column 8, lines 15-23); thus, Motomura discloses that the functional unit performs a register operation specified in the instructions in each of the plurality of threads

Hence, claim is anticipated by Motomura.

Referring to claims 15-16: Any request from the user, such as keyboard input or printing, trigs a threaded operation. Such request is an I/O operation, and the device receives the request, such as a keyboard or a mouse, is an I/O device.

Referring to claim 17: Motomura discloses the message content (column 8, lines 36-39).

Referring to claim 18: Once each thread's program counter finishes every associated instruction and is ready for a new thread, the means to ask for new thread's information indicates the completion of the previous thread.

Referring to claim 19: Motomura discloses the processing parameters and the pointer pointing to the parameters (column 8, lines 37-38). The pointer is the data register address.

Referring to claim 20: Motomura's bus is bi-directional (figure 1, structures 111 and 112, column 8, line 16).

Referring to claims 21-22: As disclosed above, Motomura discloses concurrently processing a plurality of threads by interleaving the plurality of threads. Motomura discloses that a fork demand is issued for interleaving different threads (column 8, lines 18-20) and places the earlier thread in a waiting state (column 8, lines 40-51) in order to process a new thread.

Motomura discloses executing a different thread while the first thread is in a waiting state (column 8, lines 40-42). Motomura's switching to a different thread is the claimed disabling the first thread. Furthermore, Motomura discloses an executing scenario which executing a program

sequentially without dividing (column 1, line 61). The disclosed executing a program without dividing is the claimed continuing to execute if in a non-wait instruction.

Referring to claim 23: Motomura discloses resuming the thread previously in a waiting state (column 2, lines 59-60).

Referring to claim 24: Motomura discloses an instruction processing unit (figure 1, structure 150) to process instructions fetched from a program memory (figure 1, structure 140); and a thread control unit (figure 16's thread execution control system) coupled to the instruction processing unit to manage initiating and termination of at least one of the plurality of threads.

Referring to claim 25: Motomura discloses a memory access unit (figure 16, structure 620) coupled to the instruction processing unit to provide access to one of a plurality of data memories (figure 5, structure 541) via a data memory switch (figure 5, structure 542), the memory access unit having a plurality of data base registers (figure 16, structure 1610), each of the data base registers corresponding to each of the threads; and a functional unit (figure 1, structure 110) coupled to the instruction processing unit to perform an operation specified in one of the instructions; and a register file (figure 16, structure 1420) having a plurality of data registers (figure 16, structure 1410), each of the data registers corresponding to each of the threads.

Referring to claim 27: Motomura discloses a plurality of data memories (figure 5, structure 541), a memory switch (figure 16, structure 620), and program memory (figure 16, structure 1610). Motomura discloses a processing slice (figure 1, structure 100) to execute a plurality of threads. Motomura's processing slice comprises functioning unit (figure 1, structure 110) to perform a register operation (column 8, lines 1-51). The requestor/generator for the

threaded operations is the peripheral device, and the means to convey the request is the peripheral bus.

As stated by both the Application Specification and the previous response Paper #13, each processing slice executes several instructions concurrently in the same clock cycle by interleaving the instructions' fetch, decode, dispatch, and execution (Paper#13, page 2, lines 6-7, Specification, page 8, lines 21-25). Motomura discloses forking other threads during execution of a certain thread (column 8, lines 15-27), which is the claimed execution, fetch, and decode in the same clock cycle.

As stated by both the Application Specification and previous response Paper #13, the claimed limitation "functional unit to perform a register operation specified in the instructions in each of the plurality of threads" directs toward performing an operation specified in the dispatched instruction and performs all operations of the instruction set that manipulate values in the data registers (Paper#13, page 2, lines 8-10, Specification, page 12, lines 5-7). Motomura discloses performing operations dispatched via the bus 111 (column 8, lines 15-23); thus, Motomura discloses that the functional unit performs a register operation specified in the instructions in each of the plurality of threads

Hence, claim is anticipated by Motomura.

Referring to claims 28-29: Any request from the user, such as keyboard input or printing, trigs a threaded operation. Such request is an I/O operation, and the device receives the request, such as a keyboard or a mouse, is an I/O device.

Referring to claim 30: Motomura discloses the message content (column 8, lines 36-39).

Referring to claim 31: Once each thread's program counter finishes every associated instruction and is ready for a new thread, the means to ask for new thread's information indicates the completion of the previous thread.

Referring to claim 32: Motomura discloses the processing parameters and the pointer pointing to the parameters (column 8, lines 37-38). The pointer is the data register address.

Referring to claim 33: Motomura's bus is bi-directional (figure 1, structures 111 and 112, column 8, line 16).

Referring to claims 34-35: As disclosed above, Motomura discloses concurrently processing a plurality of threads by interleaving the plurality of threads. Motomura discloses that a fork demand is issued for interleaving different threads (column 8, lines 18-20) and places the earlier thread in a waiting state (column 8, lines 40-51) in order to process a new thread.

Motomura discloses executing a different thread while the first thread is in a waiting state (column 8, lines 40-42). Motomura's switching to a different thread is the claimed disabling the first thread. Furthermore, Motomura discloses an executing scenario which executing a program sequentially without dividing (column 1, line 61). The disclosed executing a program without dividing is the claimed continuing to execute if in a non-wait instruction.

Referring to claim 36: Motomura discloses resuming the thread previously in a waiting state (column 2, lines 59-60).

Referring to claim 37: Motomura discloses an instruction processing unit (figure 1, structure 150) to process instructions fetched from a program memory (figure 1, structure 140); and a thread control unit (figure 16's thread execution control system) coupled to the instruction processing unit to manage initiating and termination of at least one of the plurality of threads.

Referring to claim 38: Motomura discloses a memory access unit (figure 16, structure 620) coupled to the instruction processing unit to provide access to one of a plurality of data memories (figure 5, structure 541) via a data memory switch (figure 5, structure 542), the memory access unit having a plurality of data base registers (figure 16, structure 1610), each of the data base registers corresponding to each of the threads; and a functional unit (figure 1, structure 110) coupled to the instruction processing unit to perform an operation specified in one of the instructions; and a register file (figure 16, structure 1420) having a plurality of data registers (figure 16, structure 1410), each of the data registers corresponding to each of the threads.

Referring to claim 41: Motomura discloses a multi-thread processor (figure 1, structure 100) having program registers (figure 16, structure 1410) and database registers (figure 16, structure 1610). Motomura discloses a processing slice (figure 1, combined structures 120, 111, 112, and 110) to execute a plurality of threads. Motomura's processing slice comprises a functioning unit (figure 1, structure 110) to perform a register operation (column 8, lines 1-51). The requestor/generator for the threaded operations is the peripheral device, and the means to convey the request is the peripheral bus.

As stated by both the Application Specification and the previous response Paper #13, each processing slice executes several instructions concurrently in the same clock cycle by interleaving the instructions' fetch, decode, dispatch, and execution (Paper#13, page 2, lines 6-7, Specification, page 8, lines 21-25). Motomura discloses forking other threads during execution of a certain thread (column 8, lines 15-27), which is the claimed execution, fetch, and decode in the same clock cycle.

As stated by both the Application Specification and previous response Paper #13, the claimed limitation "functional unit to perform a register operation specified in the instructions in each of the plurality of threads" directs toward performing an operation specified in the dispatched instruction and performs all operations of the instruction set that manipulate values in the data registers (Paper#13, page 2, lines 8-10, Specification, page 12, lines 5-7). Motomura discloses performing operations dispatched via the bus 111 (column 8, lines 15-23); thus, Motomura discloses that the functional unit performs a register operation specified in the instructions in each of the plurality of threads.

Hence, claim is anticipated by Motomura.

Claim Rejections - 35 USC § 103

3. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains.. Patentability shall not be negatived by the manner in which the invention was made.

4. The factual inquiries set forth in *Graham* v. *John Deere Co.*, 383 U.S. 1, 148 USPQ 459 (1966), that are applied for establishing a background for determining obviousness under 35 U.S.C. 103(a) are summarized as follows:

1. Determining the scope and contents of the prior art.
2. Ascertaining the differences between the prior art and the claims at issue.
3. Resolving the level of ordinary skill in the pertinent art.
4. Considering objective evidence present in the application indicating obviousness or nonobviousness.

5. Claims 13, 26, and 39 are rejected under 35 U.S.C. 103(a) as being unpatentable over the combination of the Motomura and Hiraoka et al. (U.S. Patent No. 5,418,917).

Referring to claims 13, 26, and 39: Motomura discloses an instruction fetch unit (figure 5, structure 150) to fetch the instructions from the program memory using a plurality of program counters (figure 15, structure 140), each program counter corresponding to each of the threads; an instruction decoder (figure 5, structure 542) and dispatcher (figure 5, structure 150) to decode the instructions and dispatch the decoded instructions to one of the memory access unit, the functional unit, and the peripheral unit. Motomura does not explicitly disclose a buffer to hold the fetched instructions.

Hiraoka discloses managing pipeline processing (abstract); Hiraoka teaches that the instruction buffer is a well-known industrial practice to track and to pipeline the processing requests (figure 1). Hence, it would be obvious to one having ordinary skill in the computer art at time Appellant made the invention to adapt Hiraoka's instruction buffer onto Motomura because Hiraoka teaches one to properly manage the pipeline processing with an instruction buffer.

6. Claim 40 is rejected under 35 U.S.C. 103(a) as being unpatentable over the combination of the Motomura and Dove et al. (U.S. Patent No. 5,938,765).

Referring to claim 40: Motomura discloses a multi-thread processor (figure 1, structure 100). Motomura discloses a processing slice (figure 1, combined structures 120, 111, 112, and 110) to execute a plurality of threads. Motomura's processing slice comprises functioning unit (figure 1, structure 110) to perform a register operation (column 8, lines 1-51). The

requestor/generator for the threaded operations is the peripheral device, and the means to convey the request is the peripheral bus.

As stated by both the Application Specification and the previous response Paper #13, each processing slice executes several instructions concurrently in the same clock cycle by interleaving the instructions' fetch, decode, dispatch, and execution (Paper#13, page 2, lines 6-7, Specification, page 8, lines 21-25). Motomura discloses forking other threads during execution of a certain thread (column 8, lines 15-27), which is the claimed execution, fetch, and decode in the same clock cycle.

As stated by both the Application Specification and previous response Paper #13, the claimed limitation "functional unit to perform a register operation specified in the instructions in each of the plurality of threads" directs toward performing an operation specified in the dispatched instruction and performs all operations of the instruction set that manipulate values in the data registers (Paper#13, page 2, lines 8-10, Specification, page 12, lines 5-7). Motomura discloses performing operations dispatched via the bus 111 (column 8, lines 15-23); thus, Motomura discloses that the functional unit performs a register operation specified in the instructions in each of the plurality of threads.

Motomura only discloses a multi-thread processor (figure 1, structure 100), Motomura does not discloses a plurality of the multi-thread processors. Dove discloses a computer system with a node structure (figure 1). Dove teaches one to increase the system processing power with the multi-node architecture. Each of Dove's nodes includes a plurality of processors (figure 1), which enables each node to concurrently process a plurality of threads. Since each of Dove's nodes is able to concurrently process a plurality of threads, Dove's node functions as a multi-

thread processor. Hence, it would have been obvious to one having ordinary skill in the computer art to adopt Dove's teaching onto Motomura because Dove teaches one the multi-node architecture to further increase the system processing power.

(10) Response to Argument

1. In response to Appellant's argument that Motomura does not disclose concurrently execute a plurality of threads by interleaving (page 11, 2nd and last paragraphs, page 12, 1st paragraph, page 17, 4th paragraph, last 4 lines): As stated by both the Application Specification and the previous response Paper #13, each processing slice executes several threads concurrently in the same clock cycle by interleaving the instructions' fetch, decode, dispatch, and execution (Paper#13, page 2, lines 6-7, Specification, page 8, lines 21-25). Thus, the alleged invention's processing slice executes several threads concurrently by placing the plurality of threads on an execution pipeline; and by interleaving the threads, the processing slice alternatively processes the instruction from different threads while placing the earlier thread in a waiting state.

Motomura discloses a processing slice (figure 1, structure 100) to execute a plurality of threads. Motomura's processing slice comprises a functioning unit (figure 1, structure 110) to perform a register operation (column 8, lines 1-51). The requestor/generator for the threaded operations is the peripheral device, and the means to convey the request is the peripheral bus. Motomura discloses interleaving threads during execution of a certain thread (column 8, lines 15-27, forking operation), which is placing a plurality of threads in one execution pipeline by

interleaving. Thus, Motomuras' processing slice executes a plurality of threads concurrently as disclosed in the Appellant's Specification.

As further stated by both the Application Specification and previous response Paper #13, the claimed limitation "functional unit to perform a register operation specified in the instructions in each of the plurality of threads" directs toward performing an operation specified in the dispatched instruction and performs all operations of the instruction set that manipulate values in the data registers (Paper#13, page 2, lines 8-10, Specification, page 12, lines 5-7). Motomura discloses performing operations dispatched via the bus 111 (column 8, lines 15-23); thus, Motomura's functional unit performs a register operation specified in the instructions in each of the plurality of threads as disclosed in the Appellant's Specification.

2. In response to Appellant's argument that Motomura's multithread executing system is not analogous to the peripheral unit (page 12, 2nd paragraph) and the prior arts on record doe not disclose a plurality of peripheral units (page 17, last paragraph): As stated in the rejection, it is the transaction requestor as the peripheral units, not the multithread executing system as alleged by the appellant been matched to the peripheral unit. Motomura discloses a parallel processor for executing plural threads; a processor is known to be a processing component in any computer system for processing requests from other I/O components in the same system; therefore, the I/O component, which submits requests to Motomura's processor, is the peripheral unit. Furthermore, other prior arts on record, such as Dove (figure 2) also evidences that both the processor and I/O devices are known components in a computer system.

3. In response to Appellant's argument that Motomura does not disclose the waiting instruction (page 14, 3rd paragraph, page 15, 2nd paragraph): As disclosed above, Motomura

discloses concurrently processing a plurality of threads by interleaving the plurality of threads. Motomura discloses that a fork demand is issued for interleaving different threads (column 8, lines 18-20) and places the earlier thread in a waiting state (column 8, lines 40-51) in order to process a new thread. Furthermore, Motomura also discloses an executing scenario which executing a program sequentially without dividing (column 1, line 61). The disclosed executing a program without dividing is the claimed continuing to execute if in a non-wait instruction.

Motomura discloses executing a different thread while the first thread is in a waiting state (column 8, lines 40-42). Motomura's switching to a different thread and putting the earlier thread in a waiting state is the claimed disabling the first thread.

For the above reasons, it is believed that the rejections should be sustained.

Respectfully submitted,

MARK H. RINEHART
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100

Justin King

Conferees:

Lynne Browne

LYNNE H. BROWNE
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100

MARK H. RINEHART
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100